Active

Project #: C-50-619 Cost share #: Rev #: 0
Center # : 10/24-6-R7555-0A0 Center shr #: OCA file #:
Work type : RES
Contract#: SP0000022 Mod #: Document : PO
Prime #: Contract entity: GTRC

Subprojects ? : N CFDA:
Main project #: PE #:

Project unit: GVU Unit code: 02.010.314
Project director(s):
GUENTER B K COMPUTING (404)853-9387

Sponsor/division names: E-SYSTEMS INC /
Sponsor/division codes: 204 / 001

Award period: 920706 to 930306 (performance) 930306 (reports)

Sponsor amount	New this change	Total to date
Contract value	40,000.00	40,000.00
Funded	40,000.00	40,000.00
Cost sharing amount		0.00

Does subcontracting plan apply ?: N

Title: DEVELOPMENT OF A DIRECTIONAL FILTERING ALGORITHMS

PROJECT ADMINISTRATION DATA

OCA contact: Brian J. Lindberg 894-4820

Sponsor technical contact

MARK DAY
(214)272-0515

E-SYSTEMS, INC.
P.O. BOX 660023
1200 JUPITER ROAD
DALLAS, TX 75266-0023

Sponsor issuing office

ANDY ANDERSON
(214)205-7459

E-SYSTEMS, INC.
P.O. BOX 660023
1200 JUPITER ROAD
DALLAS, TX 75266-0023

Security class (U,C,S,TS) : U ONR resident rep. is ACO (Y/N): N
Defense priority rating : N/A N/A supplemental sheet
Equipment title vests with: Sponsor GIT
NONE PROPOSED OR ANTICIPATED.
Administrative comments -
INITIATION OF PROJECT C-50-619.

GEORGIA INSTITUTE OF TECHNOLOGY
OFFICE OF CONTRACT ADMINISTRATION

NOTICE OF PROJECT CLOSEOUT

Closeout Notice Date 07/07/94

Project No. C-50-619 Center No. 10/24-6-R7555-0A0

Project Director GUENTER B K School/Lab GVU

Sponsor E-SYSTEMS INC/

Contract/Grant No. SP0000022 Contract Entity GTRC

Prime Contract No.

Title DEVELOPMENT OF A DIRECTIONAL FILTERING ALGORITHMS

Effective Completion Date 930306 (Performance) 930306 (Reports)

Closeout Actions Required:	Y/N	Date Submitted
Final Invoice or Copy of Final Invoice	Y	
Final Report of Inventions and/or Subcontracts	Y	
Government Property Inventory & Related Certificate	N	
Classified Material Certificate	N	
Release and Assignment	N	
Other	N	

Comments

Subproject Under Main Project No.

Continues Project No.

Distribution Required:

Project Director	Y
Administrative Network Representative	Y
GTRI Accounting/Grants and Contracts	Y
Procurement/Supply Services	Y
Research Property Managment	Y
Research Security Services	N
Reports Coordinator (OCA)	Y
GTRC	Y
Project File	Y
Other	N
	N

NOTE: Final Patent Questionnaire sent to PDPI.

C 50-619
6, 7, 8

Brian K. Guenter

Assistant Professor

College of Computing

Georgia Institute of Technology

Atlanta GA 30332-0280

(404) 853-9387

November 11, 1992

Steve Cooper
E-systems

Steve;

We have completed the isotropic filtering code and have used it to antialias a texture mapped version of the images you sent us. The images you will see on the tape were created by tiling a plane with copies of your original image and then viewing the plane from a user selectable viewpoint. This introduces very high spatial frequencies in the image at the horizon line. The code computes the Jacobian relating screen space pixels to texture pixels and then uses the entries in the Jacobian to compute the maximum compression factor in any direction. This determines the two levels in the pyramid to access. Each of the two pyramid levels are accessed using a piece wise cubic interpolation filter and then a linear interpolation between the two pyramid levels gives the final result. The isotropic filtering method compares very favorably in image quality with the display program you sent us for both decimation and interpolation.

We will be finished with the directional filtering code in the next week or so. The directional filtering will dramatically improve sharpness in our test images and reduce aliasing artifacts as well. As soon as we get it finished we will ship out more images antialiased with the new directional filtering method, along with images generated using the isotropic filtering so that you can compare the two.

The tape also has text files which describe the overall structure of the software we will be sending you in the next few weeks.

Brian Guenter

High Quality Image Warp Filters on Pyramids

Jack Tumblin, Brian Guenter
Georgia Institute of Technology
College of Computing
Graphics, Visualization, and Usability Center
Atlanta, Georgia 30330

ABSTRACT

Directional filters applied to prefiltered input image pyramids strike a good balance between fast computation and sharp, alias- free images, and are widely used for texture mapping and image warping. However, most published methods use implicit conversions between discrete and continuous images. This causes two flaws: 1) an input reconstruction filter is missing, causing unpredictable warp-dependent distortions and aliasing artifacts, and 2) address calculations are needlessly complex and inflexible. By including this filter we find a simpler, more accurate, and more general implementation of filtering for texture maps and image warping that can use any desired filter kernel. Effects of cascaded filters in image pyramids are also briefly discussed.

INTRODUCTION

Image pyramids are popular for texture mapping or image warping because they produce good quality image sequences efficiently, but commonly used filtering methods sacrifice some image sharpness and detail to reduce aliasing artifacts and computational cost. For some applications, such as photogrammetry, Geographical Information Systems (GIS), medical imaging and aerial photography, loss of image information is unacceptable. In these applications, users must maintain the maximum possible amount of image detail with minimal aliasing artifacts, and are willing to spend extra computing time to achieve it. This paper describes an improved method for such high-quality texture mapping or image warping tasks based on analysis in the frequency domain.

THEORETICAL MODEL OF DISCRETE IMAGE WARPS

Texture mapping or warping (we will use the latter term) is simple for functions of continuous (non-discrete) variables; it is just a matter of algebra. A 2-D continuous input image I(x,y) mapped to a 2-D continuous output image O(s,t) is completely described by warp functions that map input space (x,y) into output space (s,t):

$$O(s,t) = I(\Gamma W_x(s,t),\ \Gamma W_y(s,t))$$

$$I(x,y) = O(W_s(x,y), W_t(x,y))$$

where

$W_s(x,y) = s$ and $W_t(x,y) = t$ are TforwardU warp functions,

and $\Gamma W_x(s,t) = x$ and $\Gamma W_y(s,t) = y$ are their inverses.

Image warping only becomes messy and confusing when implemented with discrete approximations to these continuous functions, since transitions between continuous and discrete domains can introduce

artifacts such as aliasing, anisotropy, ringing, and blurring. Precise descriptions of image warping must be done in the continuous domain, since discrete images are undefined for non-integer positions. Warping requires that each discrete image must be regarded as an unambiguous representation of a continuous image. If we assume the discrete and continuous domains are related by linear, shift-invariant systems, then discrete to discrete image warping operations can be described in four steps;

1) Discrete-to-continuous conversion: convert the discrete input image IN(m,n) to continuous input image I(x,y) by convolution with an input reconstruction filter $h_{in}(x,y)$, using discrete sampling periods of T_m, T_n in x,y respectively:

$$I(x,y) = \sum_m \sum_n IN(m,n) \cdot h_{in}(x - mT_m, y - nT_n) \qquad (1)$$

Since $h_{in}(x,y)$ will overlap the boundaries of the input image, IN(m,n) must be defined for all possible m,n values. Popular methods to define these ToutsideU pixel values are

a) use a constant, such as zero or the mean value of the image
b) clip the m,n values so that $0 \le m,n \le mmax,nmax$
c) use reflection,rotation, or other means to repeat the image periodically,
d)use a local mean or other neighborhood value.

2) Warping the continuous input space (x,y) to continuous output space(s,t), as before:

$$O(s,t) = I(IW_x(s,t),\ IW_y(s,t)) \qquad (2)$$

$$I(x,y) = O(W_s(x,y), W_t(x,y)) \qquad (3)$$

3) Prefilter to avoid aliasing. Convolve the continuous output image O(s,t) with a continuous prefilter $h_{out}(s,t)$ to limit the bandwidth sufficiently to avoid noticeable aliasing in step 4.

4) Continuous-to-discrete conversion; Sample the filtered continuous output image at points arranged in a rectangular grid aligned with the s,t axes, where grid spacing is T_i, T_j respectively. Combining step 3 and 4, we have:

$$OUT(i,j) = \int\int O(iT_i - s,\ jT_j - t) \cdot h_{out}(s,t)\, ds\, dt \qquad (4)$$

IMPLEMENTATION I: THE MISSING INPUT PREFILTER

Since computers cannot store and manipulate infinite continuous images directly, the image warping steps 1-4 must be converted into operations on discrete images. This can be confusing, and we will show that step 1 is often neglected or improperly combined with step 3.

Most published methods using input image pyramids compute output pixels consecutively using Tinverse mappingU [WOHLBERG], which essentially implements steps 1-4 in reverse order. An output pixelUs sampling position is inverse-warped into input space, and its value is calculated as a weighted sum of the neighboring input pixels, where the TneighborhoodU is defined by local properties of the inverse warping functions $IW_x(s,t)$, $IW_y(s,t)$, the $h_{in}(x,y)$ filter, and the $h_{out}(s,t)$ filter.

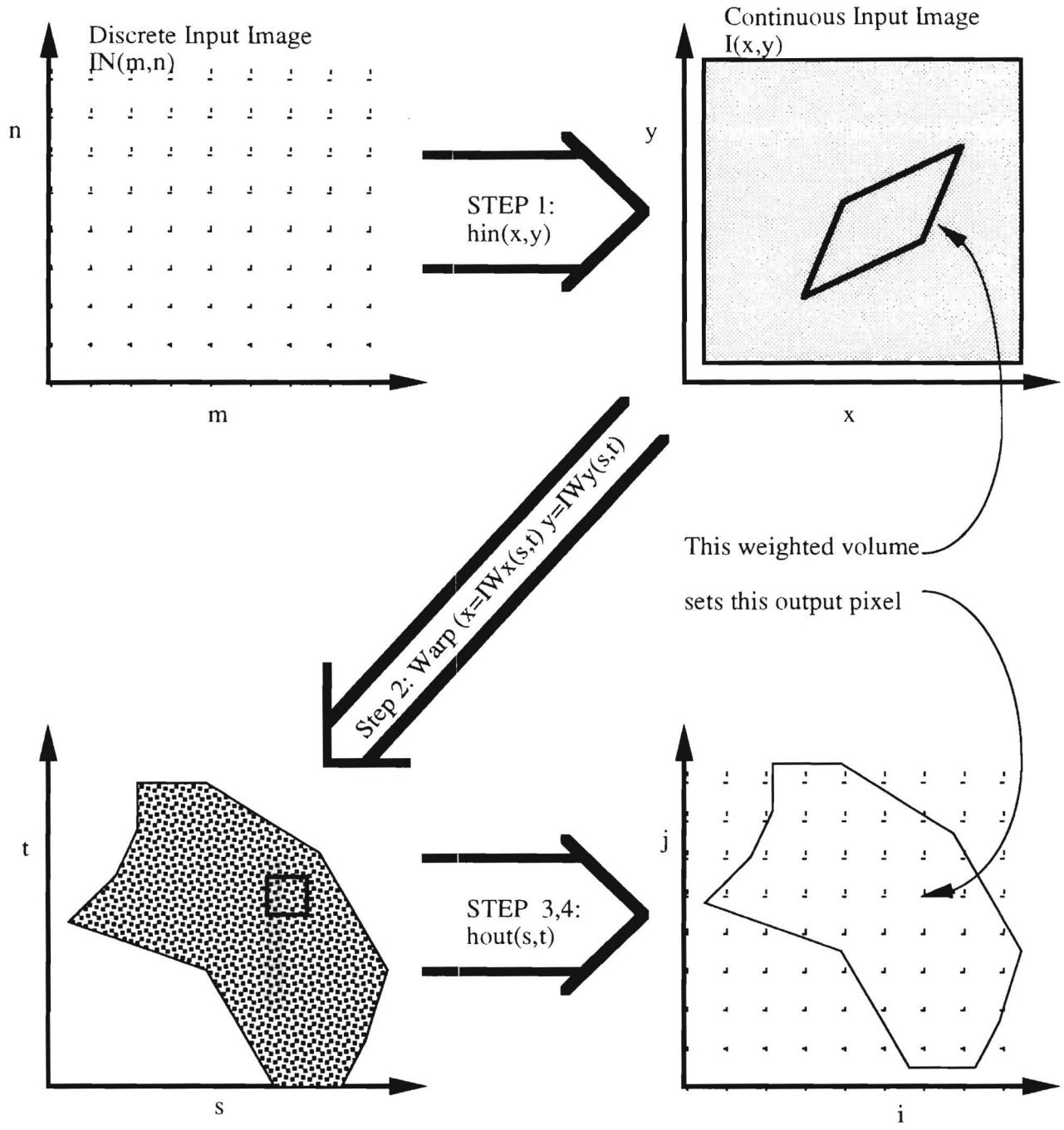


Figure 1

Though steps 1-4 include discrete representations for the input an output images, the inverse warp functions given are continuous. Inverse warping functions often cannot be expressed analytically, so storage as a discrete array is needed. Since most are locally smooth, linear approximations to warp functions are usually acceptable. This allows inverse warp functions to be stored as a discrete image in output space (i,j), where each pixel holds the corresponding continuous input image position IW(m,n) and its four partial derivatives. Expressed in matrix form, these partial derivatives are known as the Jacobian, **J**. Then the continuous warping functions at pixel position (s_0, t_0) are approximated by:

$$\mathbf{J} \equiv \begin{bmatrix} \frac{\partial \mathrm{IW}_x}{\partial s} & \frac{\partial \mathrm{IW}_x}{\partial t} \\ \frac{\partial \mathrm{IW}_y}{\partial s} & \frac{\partial \mathrm{IW}_y}{\partial t} \end{bmatrix}; \qquad \begin{bmatrix} \mathrm{IW}_x(s_0+\Delta s, t_0+\Delta t) \\ \mathrm{IW}_y(s_0+\Delta s, t_0+\Delta t) \end{bmatrix} \cong \begin{bmatrix} \mathrm{IW}_x(s_0, t_0) \\ \mathrm{IW}_y(s_0, t_0) \end{bmatrix} + [\mathbf{J}] \begin{bmatrix} \Delta s \\ \Delta t \end{bmatrix} \qquad (5)$$

Curvature and occlusion are two major sources of error in this approximation. This equation uses the first two terms of the Taylor series, so warp function curvature described by the higher-order series terms are missing. However, most warp functions are slow- varying, so these terms are quite small. More importantly, some warp functions are discontinuous, such as warps that make images of 3D self-occluding surfaces. Sampling such warp functions can cause aliasing artifacts, since it has infinite bandwidth. For critical applications, warp functions should be prefiltered (as in step 3) before they are sampled and stored as a discrete image, or used with some sort of boundary representation.

In the Tinverse mappingU implementation of image warping each output pixel value is found by integrating a weighted region of the continuous input image; the regions and weightings are derived by a reversal of steps 1-4. Given an output pixel (i,j), its value is found by substituting eqn(2) into eqn (4):

$$\mathrm{OUT}(i,j) = \int\int \mathrm{I}\left(\begin{bmatrix} \mathrm{IW}_x(i \cdot T_i - s, j \cdot T_j - t) \\ \mathrm{IW}_y(i \cdot T_i - s, j \cdot T_j - t) \end{bmatrix} \right) \cdot \mathrm{h}_{\mathrm{out}}(s,t)\, ds\, dt$$

Using the linear approximation of Eqn (5), we get:

$$\mathrm{OUT}(i,j) = \int\int \mathrm{I}\left(\begin{bmatrix} \mathrm{IW}_x(i \cdot \mathrm{T}_i, j \cdot \mathrm{T}_j) \\ \mathrm{IW}_y(i \cdot \mathrm{T}_i, j \cdot \mathrm{T}_j) \end{bmatrix} - [\mathrm{J}] \begin{bmatrix} s \\ t \end{bmatrix} \right) \cdot \mathrm{h}_{\mathrm{out}}\left(\begin{bmatrix} s \\ t \end{bmatrix} \right) ds\, dt \qquad (6)$$

This integral finds the volume under a surface over a region in continuous input space (See Figure 1). One way to imagine this process is to consider that for each output pixel (i,j), the prefilter $\mathrm{h}_{\mathrm{out}}(s,t)$ is positioned in the continuous output space, centered at that pixelUs sampling position. Then this continuous filter is inverse-warped to continuous input space (x,y), where it is used as a weighting function for the continuous input image. If the non-zero extent of the prefilter $\mathrm{h}_{\mathrm{out}}(s,t)$ is rectangular, then the linear warping approximation of equation (5) maps it to a parallelogram in continuous input space (x,y). Then the output pixel (i,j)Us value is the integral of the product of the warped output prefilter and the continuous input image.

It is very important to notice that this is a continuous integral, finding the volume in s,t space of the product of two *continuous* functions $\mathrm{h}_{\mathrm{out}}(s,t)$ and $\mathrm{I}([\mathrm{IW}(s,t)])$. Clearly, using the discrete function IN(m,n) in place of the continuous function I() is not equivalent. However, methods published by [BLINN76], [FIEBUSH80], [WILLIAMS83], [BURT], [HECKBURT86], [WOHLBERG90] and others do precisely that; they omit the discrete to continuous conversion of step 1. Although Wohlberg also presents steps 1-4, step 1 is not included in the implementations he describes. Instead, all these implementations use either the weighted sum of pixels in the *discrete* input image IN(m,n), or use pixels

chosen from an input image pyramid, perhaps assuming that the cascade of discrete filters and decimators used to build the pyramid will suffice an input reconstruction filter. This is incorrect, because input pyramids are still discrete and have a periodic spectrum, forcing the warped output prefilter to act as the input reconstruction filter as well.

EFFECT OF THE MISSING FILTER

Warped output filters can act as reasonably good input reconstruction filters, but only if they are subject to restrictions that degrade the output image quality. The reason for these restrictions are clear in the frequency domain. If no reconstruction filter is used, the conversion of the discrete image IN(m,n) to the continuous image I(x,y) creates a grid of pixel-weighted impulse functions. The 2D continuous Fourier transform is a periodic function, as shown in Fig 2, consisting of a TbasebandU spectrum centered at the origin, and spectral replicas centered at multiples of the discrete sampling frequencies $(1/T_m, 1/T_n)$.

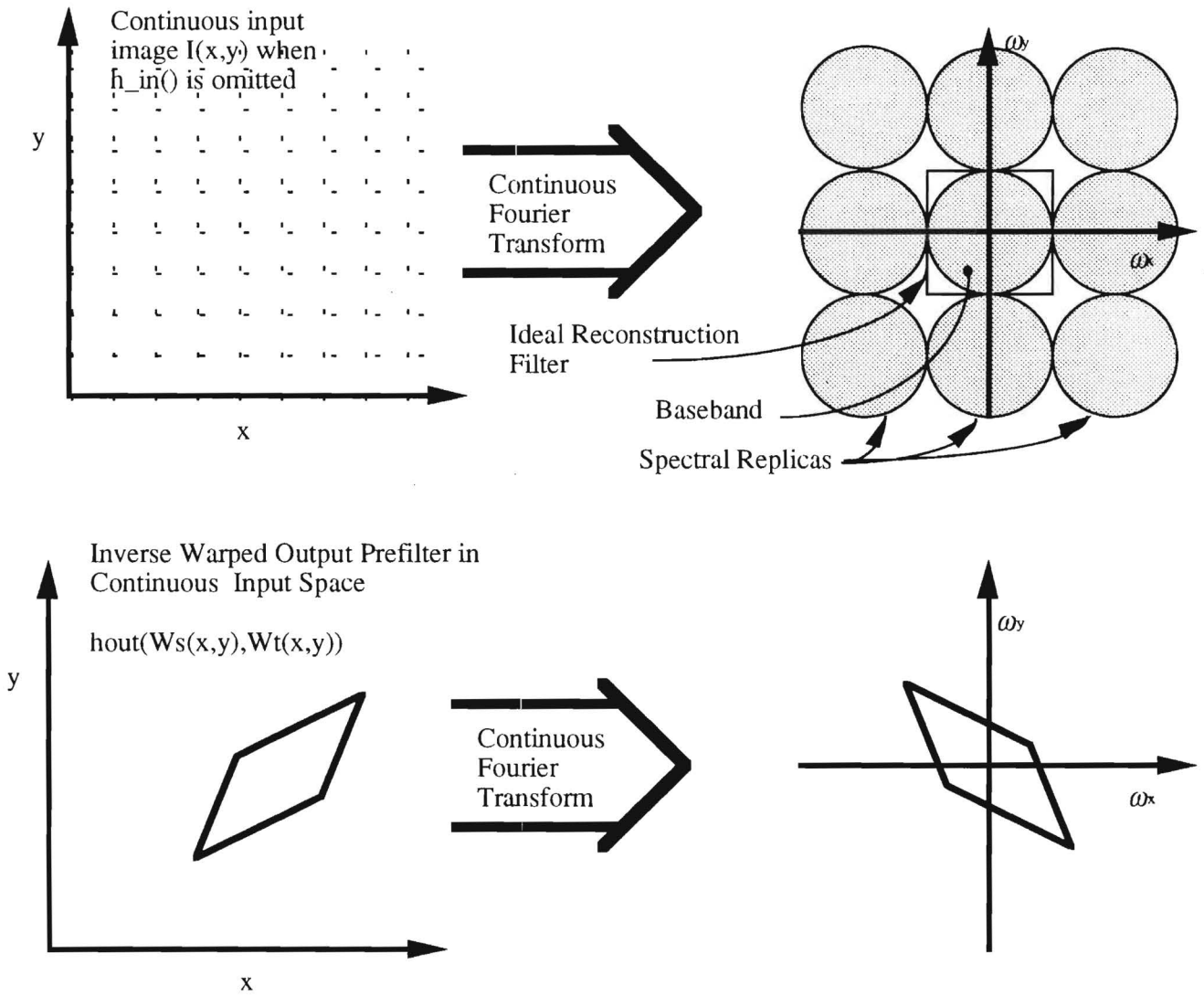


Figure 2: Output Prefilter h_in warped to continuous input space (x,y) is often ill-suited for use as the input reconstruction filter h_in().

The purpose of the input reconstruction filter is to suppress all the spectral replicas, but without degrading the baseband spectrum that describes the continuous image I(x,y). The warped output prefilter is ill-suited for this purpose. Suppose the (unwarped) output prefilter is a high-quality lowpass filter, whose non-zero extent forms a square in output frequency space (ω_s, ω_t). Due to the linear approximations of Equation (5), this extent becomes a parallelogram in the continuous input frequency space [DUDGEON84]. For maximum detail and sharpness, all of the baseband input spectrum that falls within this parallelogram must be included in the output image; but to act as a good input prefilter, the warped output filter must also avoid overlapping any of the spectral replicas--an impossible goal for a parallelogram. This forces restrictions that must reduce image quality.

[BLINN76],[WILLIAMS83] and others constrain the inverse-warped prefilterUs spatial extent to be square, causing severe blurring when warp functions are strongly anisotropic (i.e. when the inverse warped output prefilter forms a long, thin parallelogram). HeckbertUs TEWA on PyramidsU method [] allows shape changes in the inverse warped output prefilter, but must limit its aspect ratio (major axis length/minor axis length) to avoid aliasing, and is restricted to circularly symmetric $h_{out}(s,t)$ filters. Burt also suggests minimum size requirements for filters, and Wohlberg notes that only space-invariant warps can be correctly implemented with a single filter.

Position -Dependent Error

Image warping without an input reconstruction filter also has unwanted effects in the spatial domain, as noted by [BURT]. Applying an inverse warped, continuous output prefilter to a discrete input image is equivalent to applying a selection from a family of discrete filters constructed by sampling the continuous filter at various subpixel positions. [BURT] noted that the discrete spectrum of a sampled continuous filter varied markedly with sampling position as the sampling rate decreased, as shown in Fig 3. For example, suppose a triangle filter TRI(s,t) is used as an output prefilter. If it is Tinverse-warpedU to continous input space by a scale factor TZoomU, and regularly sampled from a starting point (xoff,yoff), it forms a family of discrete filters Fdiscr(m,n,xoff,xoff);

TRI(s,t) = (1-|x|)*(1-|y|) for |x|,|y| < 1; =0 otherwise
Fdiscr(m,n,xoff,yoff) = TRI(Zoom*(m-xoff),Zoom*(n-yoff))

If no input reconstruction filter is used, the effect of applying the warped output filter is equivalent to applying members of the Fdiscr() family of discrete filters according to the output pixelUs subpixel position xoff,yoff in input space. Figure 3 shows the discrete spectrum of one axis of Fdiscr() for Zoom= 1, 1.5, 2.0 and 2.5. Note that for Zoom<2 the discrete filterUs spectrum is strongly dependent on subpixel offset xoff, and at most subpixel positions is a poor choice for supression of spectral replicas occuring above pi radians/sample. We found similar results for the cubic spline filter [MITCHELL88], and others.

IMPLEMENTATION II: ADDRESSING AND THE INTEGRATION GRID

The filter restrictions or the position-dependent errors that arise from neglecting the input reconstruction filter are unacceptable for high-quality image warping; we must calculate the continuous integral of equation (6). Paradoxically, this integral helps simplify and generalize the warping implementation.

Without the input reconstruction filter of step 1), finding the value of an output pixel requires sophisticated address generation, since these methods must find a weighted sum of all discrete input pixels that fall within the extent of the output prefilter, after it is warped into input space. The linear approximations of equation (6) will warp a rectangular output prefilter into an arbitrary parallelogram, and generating all the input pixel addresses within this region is rather complicated. [HECKBERT86] solved this by using a circular output filter extent; the corresponding ellipse in input space encloses pixels whose addresses generated by a quadratic incremental method, but restricts users to circularly symmetric output prefilters. None of this is necessary when equation (6) is implemented.

Equation (6) can be evaluated by almost any numerical integration method; we chose piecewise-constant approximations. Take a regular grid of point samples of the function to be integrated, and let each sample approximate the functionUs value over a small local area. The integral is approximately equal to a weighted sum of those samples. An integration grid aligned with the the major and minor axes of a parallelogram-shaped area is particularly easy to compute, and since all samples represent equal areas, all will receive the same weighting function value. Incremental methods (DDAUs) can find both the input and output-space addresses with as little as 3 add operations per grid position, and in output space the integration grid is aligned with s,t axes; thus separable filters are especially efficient to compute.

$$\mathrm{OUT}(i,j) = \int\int \mathrm{I}\left(\begin{bmatrix} \mathrm{IW}_x\left(i\cdot \mathrm{T}_i,\, j\cdot \mathrm{T}_j\right) \\ \mathrm{IW}_y\left(i\cdot \mathrm{T}_i,\, j\cdot \mathrm{T}_j\right) \end{bmatrix} - [\mathrm{J}]\begin{bmatrix} s \\ t \end{bmatrix}\right) \cdot \mathrm{h}_{\mathrm{out}}\left(\begin{bmatrix} s \\ t \end{bmatrix}\right) ds\, dt$$

$$\cong \sum_p \sum_q \mathrm{I}\left(\begin{bmatrix} \mathrm{IW}_x\left(i\cdot \mathrm{T}_i,\, j\cdot \mathrm{T}_j\right) \\ \mathrm{IW}_y\left(i\cdot \mathrm{T}_i,\, j\cdot \mathrm{T}_j\right) \end{bmatrix} - [\mathrm{J}]\begin{bmatrix} p\cdot sstep \\ q\cdot tstep \end{bmatrix}\right) \cdot \mathrm{h}_{\mathrm{out}}\left(\begin{bmatrix} p\cdot sstep \\ q\cdot tstep \end{bmatrix}\right) \qquad (7)$$

where sstep,tstep == integration grid spacing in continuous output space s,t;
p,q == integration grid indices that span the non-zero extent of the output filter $\mathrm{h}_{\mathrm{out}}$
N == number of integration grid points

We can find the spacing required for the integration grid by restating the piecewise-constant approximation in equation (7) above as a pair of image resampling problems, and exploring them in the frequency domain. This integral involves two TimagesU that are parallelogram-shaped in continuous input space and rectangular-shaped if transformed to output space. These are 1) the Tfilter subimageU, defined as the non-zero extent of the output filter $\mathrm{h}_{\mathrm{out}}()$, and 2) the Tinput subimageU defined as the region of the continuous input image I() covered by the filter subimage. Both continuous subimages can be exactly represented by the samples taken by the integration grid, but only if both subimages have finite bandwidth, and if the integration grid meets the Nyquist Criterion for both subimages (i.e. sampling frequencies are at least twice the cutoff frequencies of the subimages). Fortunately, both conditions can always be met; both subimage bandwidths are limited, and our choice of integration grid has no other constraints.

Choosing the integration grid spacing is easiest to understand in continuous output space(s,t) and in its spatial frequency domain (ω_s, ω_t). Since the integration grid is aligned with the s,t axes, maximum spatial frequencies of the product of the subimages in this space will inversely set the integration grid spacings. The integration grid samples of a continuous signal in s,t space will be an unambiguous representation of that signal only if sstep and tstep are chosen so that the signalUs highest spatial

frequencies are less than the Nyquist limit of $\frac{1}{2 \cdot sstep}, \frac{1}{2 \cdot tstep}$.

Convert both the input and the filter subimages to continuous output space s,t; then find the spectrum of their product to determine sstep and tstep. In s,t space, the filter subimage is simply the output prefilter h_{out}. Since $h_{out}()$ limits bandwidth for output sampling (step 3 above), the non-zero extent of its spectrum in the frequency domain is a parallelogram (actually a rectangle) centered at the origin, with major/minor axes given by

$$(\omega_s, \omega_t) = \left(\frac{1}{2 \cdot T_i}, 0\right) \text{ and } \left(0, \frac{1}{2 \cdot T_j}\right).$$

The input subimage is similarly bandlimited in the input space x,y. Since it is reconstructed from the discrete input image IN() using an sampling axes of $(T_m, 0)$ and $(0, T_n)$, its highest spectral components are limited to $\left(\frac{1}{2 \cdot T_m}, 0\right)$, and $\left(0, \frac{1}{2 \cdot T_n}\right)$ in x,y space. The linear warp approximation of Eqn (5) transforms these input sampling axes to form output-space (s,t) sampling axes $\overline{xx}$ and $\overline{yy}$ respectively, where

$$[\overline{xx} : \overline{yy}] \equiv \begin{bmatrix} xx_s & yy_s \\ xx_t & yy_t \end{bmatrix} = [\mathbf{J}]^{-1} \begin{bmatrix} T_m & 0 \\ 0 & T_n \end{bmatrix} \qquad (8)$$

Accordingly, the spectrum of the input subimage in output space is a parallelogram with major/minor axes given by $\left(\frac{1}{2 \cdot xx_s}, \frac{1}{2 \cdot xx_t}\right)$ and $\left(\frac{1}{2 \cdot yy_s}, \frac{1}{2 \cdot yy_t}\right)$.

Integration grid spacing small enough to represent the two sub-images are insufficient to represent their product, as needed in equation (7). Multiplication of two subimages in the spatial domain is equivalent to convolution of their spectra in the frequency domain, so integration grid spacing must increase to avoid aliasing of a product spectrum that is no longer a simple parallelogram. However, the eight sided product spectrum will always fall within a rectangular region of the ω_s, ω_t plane bounded by

$$|\omega_s| < \frac{1}{2 \cdot T_i} + \frac{1}{2 \cdot xx_s} + \frac{1}{2 \cdot yy_s} \quad \text{and} \quad |\omega_t| < \frac{1}{2 \cdot T_j} + \frac{1}{2 \cdot xx_t} + \frac{1}{2 \cdot yy_t}$$

shown graphically in Figure XX. This means that the integration grid spacing should be

$$sstep \le \frac{1}{\frac{1}{T_i} + \frac{1}{xx_s} + \frac{1}{yy_s}} \qquad \text{and} \quad tstep \le \frac{1}{\frac{1}{T_j} + \frac{1}{xx_t} + \frac{1}{yy_t}} \qquad (9)$$

to avoid aliasing in evaluation of Equation (7). Output pixels where the warping function greatly compresses the input image will require an unreasonably large number of integration grid points to avoid undersampling the input image. An input image pyramid can greatly reduce the number of integration grid points needed.

IMPLEMENTATION III; USING INPUT IMAGE PYRAMIDS

Prefiltered image pyramids, as described by previous workers, create a hierarchy of discrete images called TlevelsU. Each level is created by discrete low-pass filtering and decimation of a previous level. Define Tlevel 0U of the pyramid as the original input image, and level n as the filtered-and-decimated image that has one row of pixels for every 2^n rows in level 0. Because each TlevelU represents a continuous image of the same size, the level 0 sampling period of T_m, T_n becomes a sampling period of $2^n T_m$, $2^n T_n$ for the level n image. Accordingly, if the continuous image represented by level 0 has a bandwidth of $\frac{1}{2 \cdot T_m}, \frac{1}{2 \cdot T_n}$, then $\frac{1}{2^{n+1} \cdot T_m}, \frac{1}{2^{n+1} \cdot T_n}$ is the approximate bandwidth of the continuous image represented by level n; thus selecting a pyramid level is roughly equivalent to selecting a bandwidth octave for the continuous input image I(). The accuracy of an image pyramidUs bandwidth limiting action depends on the quality of the filters used and the cumulative effects of cascading them with decimators.

Input image pyramids are easily accommodated in equations (7,9) by two steps:
1)include appropriate scaling for the input sampling vectors; replace equation (8) with:

$$[\overline{xx} : \overline{yy}] = \begin{bmatrix} xx_s & yy_s \\ xx_t & yy_t \end{bmatrix} = [\mathbf{J}]^{-1} \begin{bmatrix} 2^{\text{level}} \cdot T_m & 0 \\ 0 & 2^{\text{level}} \cdot T_n \end{bmatrix}$$

2) devise a method to select the pyramid level. Since increased pyramid level decreases input image bandwidth, select the highest possible level that will not discard frequency components that pass through the output prefilter, $h_{out}()$. This is best done in continuous input spaceUs frequency domain. Here, the bandwidth of the pyramid level is a rectangle bounded by

$$\left(|\omega_x|, |\omega_y|\right) \le 2^{-level} \cdot \left(\frac{1}{2 \cdot T_m}, \frac{1}{2 \cdot T_n}\right), \text{ and the}$$

output prefilterUs bandwidth is a parallelogram centered at the origin with major/minor axes given by $(1/2\,ss_x,\ 1/2\,ss_y)$ and $(1/2\,tt_x,\ 1/2\,tt_y)$ where

$$[\overline{ss} : \overline{tt}] \equiv \begin{bmatrix} ss_x & tt_x \\ ss_y & tt_y \end{bmatrix} = [\mathbf{J}] \begin{bmatrix} T_i & 0 \\ 0 & T_j \end{bmatrix} \qquad (11)$$

A bounding rectangle for this parallelogram has bandwidth of $(|1/(2\,ss_x)|+|1/(2\,tt_x)|,\ |1/(2\,ss_y)|+|1/(2\,tt_y)|)$. To guarantee full output bandwidth, choose highest level of the pyramid that completely encloses the warped prefilterUs parallelogram extent; using the bounding rectangle the condition for 'level' is

$$\left(\left|\frac{1}{2 \cdot ss_x}\right| + \left|\frac{1}{2 \cdot tt_x}\right|,\ \left|\frac{1}{2 \cdot ss_y}\right| + \left|\frac{1}{2 \cdot tt_y}\right|\right) \le 2^{-level} \cdot \left(\frac{1}{2 \cdot T_m}, \frac{1}{2 \cdot T_n}\right)$$

or equivalently,

$$\text{level} \le -\log_2\left(\text{MAX}\left\{\left|\frac{T_m}{ss_x}\right| + \left|\frac{T_m}{tt_x}\right|,\ \left|\frac{T_n}{ss_y}\right| + \left|\frac{T_n}{tt_y}\right|\right\}\right) \qquad (12)$$

Thus high quality texture mapping with arbitrary hin() and hout() filters applied to an input image pyramid can be implemented by following equations 11, 12, 10, 9, and 7.

Perils and Practical Limits

Using pyramids makes the calculation of these equations tractable. We found it is also advisable to:

a) limit the minimum and maximum size of the partial derivatives to set bounds on the number of integration grid points calculated per pixel,

b) define values for all pixel addresses for all levels of the pyramid, since integration grids may extend beyond the edge of the image;

c) create the image pyramids carefully to avoid distortions.

Image pyramids for high quality image warping must be built with care, since repeated filtering and decimation can cause significant distortion of the input signal by both aliasing and cumulative filter errors. Each level of an image pyramid is built by filtering the previous level and decimating its pixels by 2x2. Pixels in the highest levels of the pyramid are related to pixels in level 0 by a weighting function that evolved from repeated discrete convolutions of the pyramid filter, but this cumulative filter may evolve in unintended ways. The Gaussian-like filter was designed by [BURT] to avoids these problems, but its poor high-frequency response causes blurring. Other filters such as [Mitchell/Netravali]Us cubic filter gives sharper images, but allow some aliasing and cumulative distortions. To avoid these problems, we found two alternatives:

a) With Gaussian pyramids [BURT], use a lower level in the pyramid where frequencies of interest are not attenuated, or

b) When using sharper filters such as [MITCHELL88], construct level n of the pyramid from level n-2 or lower to reduce aliasing and cumulative distortions to the filter.

RESULTS

Image plates show the results of the image warping method described above. Plate 1 uses a warp function which forms a 3D perspective-projection view of an infinite plane covered with repetitions of the input image. The input image is a checkerboard, intentionally off-center so one edge is visible at joints on the output image's infinite plane. This image warping was done using the non-directional TMIP-MAPU method described by [WILLIAMS83], where the warped input image is constrained to be square in input space, but using the cubic-spline filter of [MITCHELL88] for both the pyramid building and level-interpolation filters. Plate 2 shows the same image from the same image pyramid, but computed using both an input interpolation filter $h_{in}()$ (also Mitchell's) and unconstrained, inverse warped output prefilter (again, Mitchell's). Note the dramatic increase in image sharpness; the checkerboard pattern is discernible almost to the opposite end of the checkerboard, where the horizontal checkerboard frequency exceeds pi radians/cycle. Plate 3 and 4 are identical to 1 and 2, except the input image is now a Tzone plateU function. A zone plate is a circularly symmetric cosinusoid function$0.5 \cdot \left(1 + \cos\left(\pi a r^2\right)\right)$whose Tinstantaneous frequencyU (1st derivative of the cosineUs argument) increases linearly from zero, so local spatial frequency is directly proportional to position, and aliasing artifacts appear as offset replicas of the baseband TbullseyeU. Note that plate 4 shows strong directional filtering effects; as the zone plate replicas approach the horizon, the TbullseyesU are still visible, along with several horizontal cycles.

CONCLUSIONS

We have shown that including an input reconstruction filter in gives several important advantages in image warping;

1) Both filter restrictions and position-dependent errors are eliminated; previous methods used inverse warped output prefilters to perform input reconstruction tasks as well, forcing compromises between image sharpness and lack of position-dependent errors, including aliasing.

2) freedom to choose integration methods. We chose simple super-sampling for speed, but more accurate methods of efficient 2D integration.

3) Simple address generation Q previous methods were required to find all discrete input pixels that fell within the extent of the output prefilter warped into input space. This membership test for local input pixels is rather complicated for arbitrary quadrilaterals in input space. Heckburt solved this by using a circular output filter extent; input pixels that fall in the corresponding ellipse in input space can be generated by a quadratic incremental method. However this restricted users to circularly symmetric output prefilters. None of this is necessary when an input reconstruction filter is used; a simple uniform sampling grid aligned with the output prefilterUs axes can be quickly calculated by incremental methods

4) No restrictions on choice of filters. Both the input reconstruction filter and output prefilters are independent of this new method. HeckburtUs EWA method for directional filtering required radially-symmetric filters.

5) Increased computational cost. For each output pixel, we require evaluation of the continuous input image over a grid of points, each point requiring an interpolation from the discrete input image. The computing cost can grow substantially, especially for highly anisotropic warping functions, but well known methods such as lookup tables and accumulation of coeffcients can help reduce this cost.
We also noted that constructing image pyramids by cascaded filters can sometimes lead to spectral distortions in the decimated images. These can be prevented by limiting the length of the filter/decimation cascades, choosing lower pyramid levels, or by choosing filters with better cascade performance.

BIBLIOGRAPHY

[BLINN76] Blinn, James F., and Martin E. Newell, "Texture and Reflection in Computer Generated Images", *Communications of the ACM*, vol. 19, no. 10, pp. 542-547

[BURT] Burt, Peter J., "The Pyramid as a Structure for Efficient Computation", PAMI? pp 6-23.

[DUDGEON84] Dudgeon, Dan E., and Russell M. Mersereau, *Multidimensional Digital Signal Processing*, Prentice-Hall Inc., Englewood Cliffs, New Jersey, 1984.

[FEIBUSH80] Fiebush, Elliot A., Marc Levoy, and Robert L. Cook, "Synthetic Texturing Using Digital Filters" *Computer Graphics--SIGGRAPH '80' Proceedings*, vol. 14, no. 3, pp. 294-301, July 1980.

[HECKBURT86] Heckburt, Paul, "Survey of Texture Mapping", *IEEE Computer Graphics and Applications*, vol 6, no. 11, pp.56-67. July 1986.

[MITCHELL88] Mitchell, Don P., and Arun N. Netravali, "Reconstruction Filters in Computer Graphics", *Computer Graphics--SIGGRAPH '88 Proceedings*, vol. 22, no. 4, pp.221-228, August 1988.

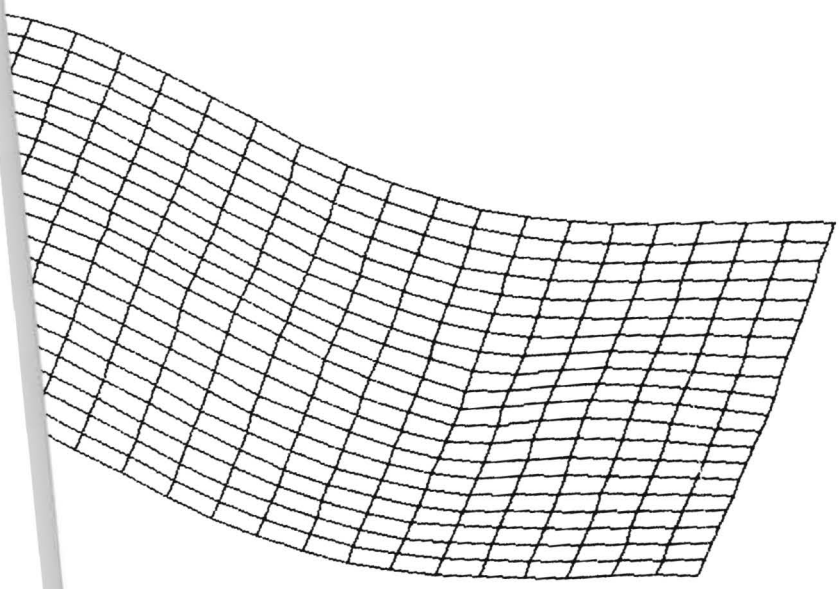
[WILLIAMS83] Williams, Lance, "Pyramidial Parametrics", *Computer Graphics--SIGGRAPH'83 Proceedings*, vol. 17, no. 3, pp1-11, July 1983.

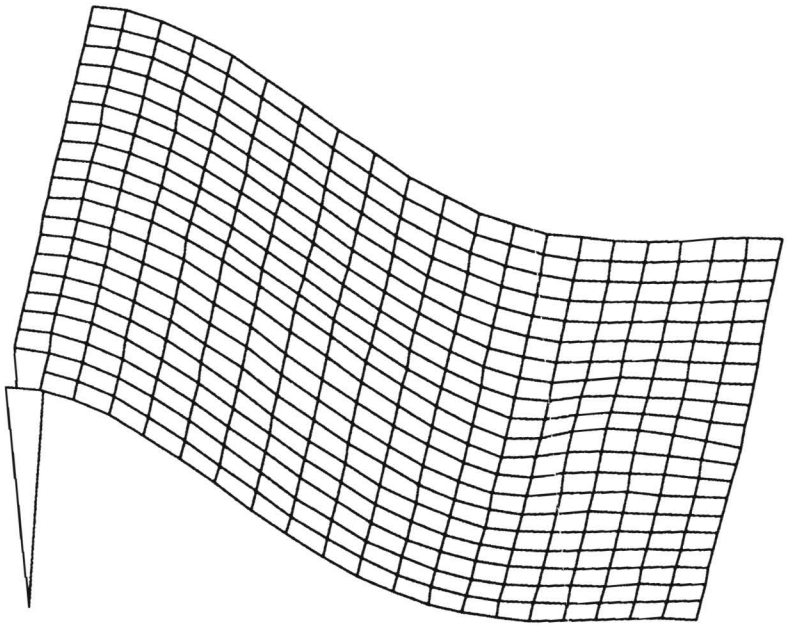
[WOHLBERG90] Wohlberg, George, Digital Image Warping, IEEE Computer Society Press, Los Alamitos, CA., 1990

filwidth = 2.5

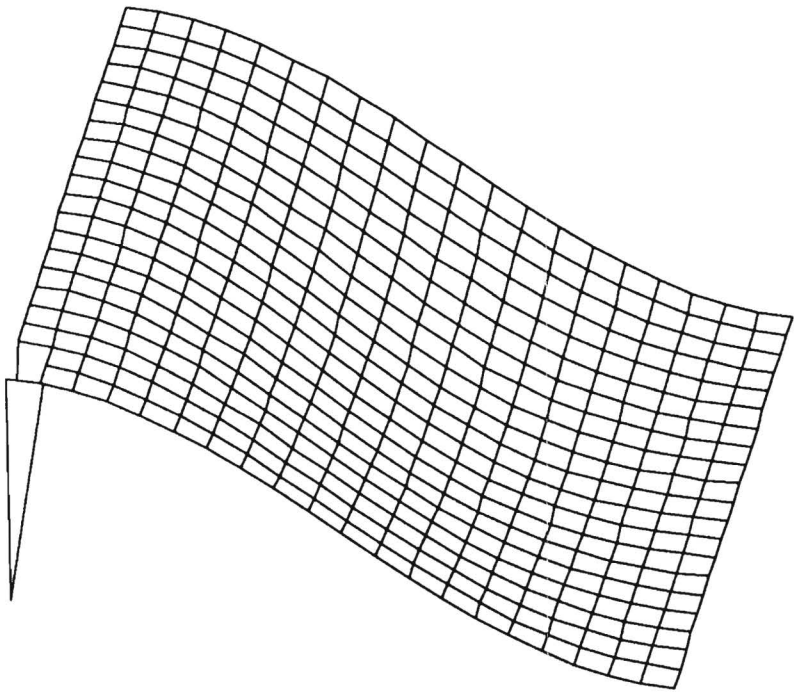
ght,)

the page):

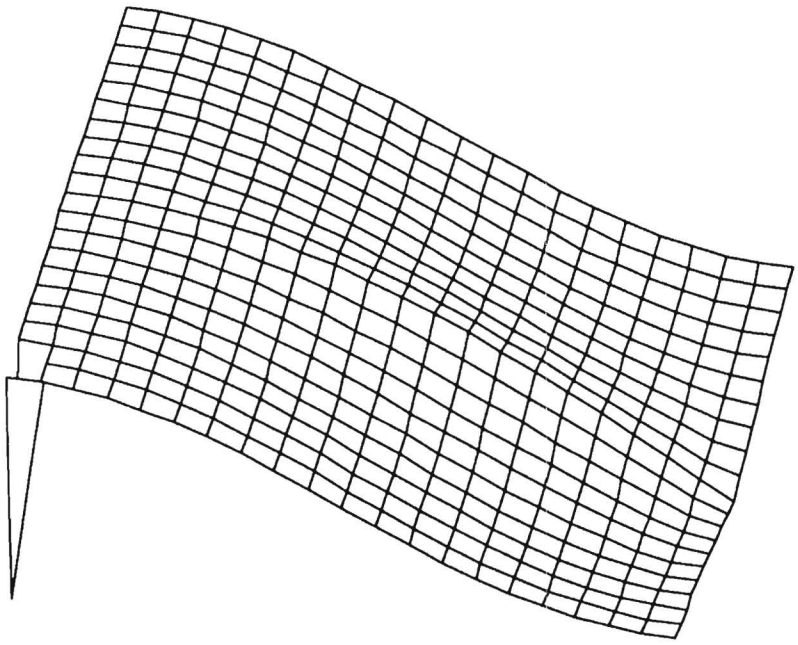




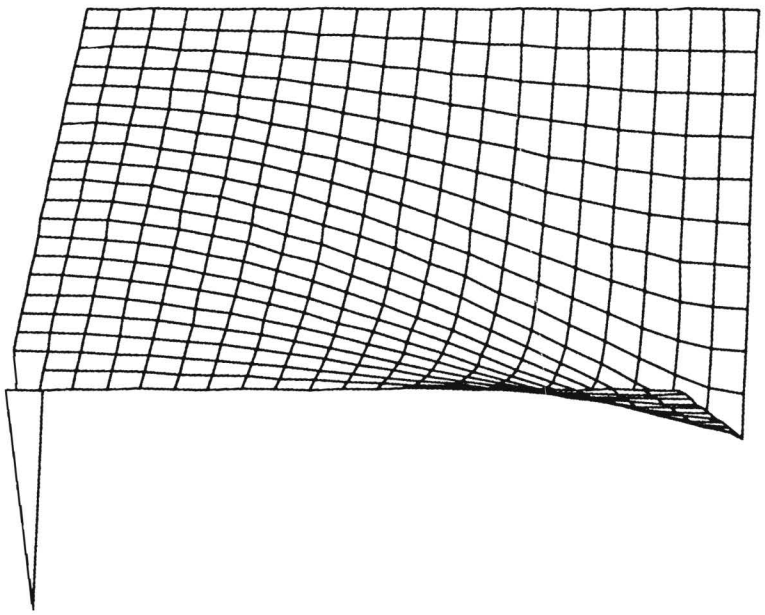
DISP



DISP



DISP



DISP